Surrogate-assisted Multiobjective Optimization based on Decomposition: A Comprehensive Comparative Analysis

Nicolas Berveglieri
Univ. Lille, CNRS, Centrale Lille, Inria, UMR 9189 - CRIStAL, F-59000 Lille, France
nicolas.berveglieri@univ-lille.fr

Bilel Derbel
Univ. Lille, CNRS, Centrale Lille, Inria, UMR 9189 - CRIStAL, F-59000 Lille, France
bilel.derbel@univ-lille.fr

Arnaud Liefooghe
Univ. Lille, CNRS, Centrale Lille, Inria, UMR 9189 - CRIStAL, F-59000 Lille, France
arnaud.liefooghe@univ-lille.fr

Hernán Aguirre
Shinshu University, Faculty of Engineering, Nagano, Japan
ahernan@shinshu-u.ac.jp

Kiyoshi Tanaka
Shinshu University, Faculty of Engineering, Nagano, Japan
ktanaka@shinshu-u.ac.jp

ABSTRACT

A number of surrogate-assisted evolutionary algorithms are being developed for tackling expensive multiobjective optimization problems. On the one hand, a relatively broad range of techniques from both machine learning and multiobjective optimization can be combined for this purpose. Different taxonomies exist in order to better delimit the design choices, advantages and drawbacks of existing approaches. On the other hand, assessing the relative performance of a given approach is a difficult task, since it depends on the characteristics of the problem at hand. In this paper, we focus on surrogate-assisted approaches using objective space decomposition as a core component. We propose a refined and fine-grained classification, ranging from EGO-like approaches to filtering or pre-screening. More importantly, we provide a comprehensive comparative study of a representative selection of state-of-the-art methods, together with simple baseline algorithms. We rely on selected benchmark functions taken from the bbob-biobj benchmarking test suite, that provides a variable range of objective function difficulties. Our empirical analysis highlights the effect of the available budget on the relative performance of each approach, and the impact of the training set and of the machine learning model construction on both solution quality and runtime efficiency.

CCS CONCEPTS

• **Theory of computation → Gaussian processes**; *Stochastic control and optimization*; Algorithm design techniques.

KEYWORDS

Multiobjective optimization, surrogates, benchmarking.

ACM Reference Format:
N. Berveglieri et al.. 2019. Surrogate-assisted Multiobjective Optimization based on Decomposition: A Comprehensive Comparative Analysis. In *Genetic and Evolutionary Computation Conference (GECCO '19), July 13–17, 2019, Prague, Czech Republic*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3321707.3321836

GECCO '19, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6111-8/19/07…$15.00
https://doi.org/10.1145/3321707.3321836

1 INTRODUCTION AND BACKGROUND

General Context. Multiobjective optimization problems (MOPs) [22, 28] aims at simultaneously optimizing two or more objectives. Given that these objectives are typically in conflict, there exist a whole set of optimal solutions that provide different quality trade-offs. The challenging task of computing a high-quality approximation set can be tackled using multiobjective optimization evolutionary algorithms (MOEAs). MOEAs were proved to be well-suited for blackbox problems, for which a mathematical formulation of the objective functions might not be available. In this paper, we are motivated by the challenges underlying the high cost of computing the objective values, e.g., when running a heavy simulation process [3]. More specifically, we are interested in an *expensive* optimization scenario [23], where the cost of computing the objective value(s) impose drastic restrictions on the number function evaluations. Depending on the CPU time that can be afforded, an algorithm designer must accommodate to a relatively short budget, typically few hundreds to few thousands evaluations. As a consequence, using standard evolutionary operators, which are supposed to serve the search in a non-expensive setting, become problematic because of the difficulty of finely controlling their stochastic behavior. In this context, adapting conventional MOEAs so that they can operate in a combined manner with well-established machine-learning techniques have attracted much attention in recent years [5, 7].

Related work overview. On the one hand, a general idea that was extensively explored so far is to build one (or multiple) surrogate model(s) from which some information about the (unknown blackbox) objective functions can be extracted and injected to the search process in order to help reaching the most interesting regions more efficiently [14]. A broad range of investigations can be found on using surrogates to assist MOEAs, such as those based on support vector regression [16], regression trees [20], radial basis functions [25], or Gaussian processes [27]. Depending on the nature of the considered surrogates, several design options might be adopted. This ranges from standard approaches where cheap surrogates are used in lieu of the expensive objective functions,

to those using the surrogates for pre-screening solutions generated by the evolution engine. Other options from single-objective Bayesian global optimization and reinforcement learning consist in a criterion to finely tune the exploitation-exploration trade-off when sampling new points. A number of merit functions and infill criteria were already investigated and combined with MOEAs for selecting the most beneficial solutions to be evaluated [10, 19].

On the other hand, despite the substantial progress achieved in the last decade, there are relatively few systematic investigations aiming at providing a better understanding of what makes a particular approach distinguishable from others, and under which optimization scenario, in terms of budget and problem difficulty, it is recommended. We can cite [5] which provides a coarse-grained taxonomy of surrogate-assisted MOEAs with a particular focus on constrained MOPs. In [7], one can also find a complementary taxonomy, and a discussion on extending existing approaches to support the production of a batch of multiple points that can be evaluated in parallel. Besides considering a simple parallel extension of ParEGO [12], using a set of weight vectors, the most advanced MOEAs studied therein fall into the class of dominance-based and indicator-based algorithms such as SMS-EGO [18]. This taxonomy came with an interesting empirical analysis of the different considered extensions with a focus on the potential gain in terms of parallelism. Nonetheless, a class of state-of-the-art EMO approaches was only considered at a small extent, namely, surrogate-assisted variants of the so-called MOEA based on Decomposition (MOEAD) [26].

The MOEAD (multiobjective evolutionary algorithm based on decomposition) framework belongs to the class of aggregation-based approaches. The original MOP is transformed into a number of (single-objective) sub-problems, defined using a scalarizing function, that are solved cooperatively. Due to its efficiency and flexibility in integrating existing single-objective optimizers into the multiobjective setting, MOEAD has become one of the most studied approaches. Nonetheless, when dealing with an expensive setting, we are not aware of any systematic analysis of what makes decomposition beneficial and how to effectively integrate surrogates in the original cooperative solving process of MOEAD. Interestingly, the most recent surrogate-assisted approaches are based on leveraging concepts from multiobjective decomposition [4, 9, 16, 25, 27]. This is precisely where the contribution of this paper lies.

Methodology and contributions overview. This paper can be viewed as the continuation of previous works on understanding and delimiting the core concepts of surrogate-assisted MOEAs [5, 7], with a particular focus on decomposition techniques. We thereby propose to discuss a fine-grained classification of existing approaches in light of the taxonomy provided in [7]. This is in an attempt to grasp the specificity of existing decomposition-based MOEAs, and to highlight the common algorithmic components that can be integrated interchangeably in different approaches. We also include simple archetype algorithms serving as baseline and which are barely considered in previous works. In fact, besides reviewing a number of distinguishable surrogate-assisted MOEAs within the framework of existing taxonomies, our goal is to shed the light on the strengths and weaknesses of existing approaches through an extensive benchmarking effort. Indeed, unfortunately, despite a skillful design, existing algorithms are often validated on different benchmark functions and under different, sometimes incomparable, settings. In our work, we aim at providing a comprehensive feedback on the relative performance of advanced approaches from the literature, especially when compared to *a priori* simple design choices. We rely on a number of problems taken from the bi-objective black-box optimization benchmarking test suite (bbob-biobj), and providing a representative sample of single-objective function combinations with well-known facets of difficulty [15]. Our empirical findings are to be considered as a step towards providing a unified view of surrogate-assisted MOEAs based on decomposition, and their relative performance under different scenarios. In particular, we study the impact of clustering when training the surrogate models, both on running time and on quality, as well as the impact of problem characteristics on convergence and running time.

Outline. In Section 2, we provide a classification of some distinguishable surrogate-assisted MOEAs. In Section 3, we report our experimental findings. In Section 4, we conclude the paper.

2 SURROGATE-ASSISTED MOEA BASED ON DECOMPOSITION (S-MOEAD)

In this section, we provide a fine-grained classification of existing surrogate-assisted MOEAs based on decomposition (S-MOEAD) in light of the taxonomy introduced in [7]. Our goal is to capture the design choices to be made when combining surrogate models with decomposition, hence eventually leveraging existing frameworks. We shall then provide a more focused description of some representative and state-of-art S-MOEAD algorithms from the literature, illustrating how the fine-grained classification captures them. The discussed algorithms will also serve to conduct our benchmarking analysis, which is the core contribution of the paper. Before going in a more detailed discussion, let us first define the multiobjective decomposition framework in the context of our work.

2.1 Decomposition-based MOEA in a Nutshell

Solving a MOP using decomposition implies to define a number of *a priori* smaller single- (or multiobjective) sub-problems. In our work, we assume a sub-problem is simply defined by aggregating the original objectives using a scalarizing function. We consider an objective function vector $F\colon \mathbb{R}^d \mapsto \mathbb{R}^m$ to be minimized, such that $\mathbb{R}^d$ is the (continuous) variable space and $\mathbb{R}^m$ is the objective space. For two solutions $x, x' \in \mathbb{R}^d$, x is dominated by x' if, for all $i \in \{1, \ldots, m\}$, $f_i(x) \leqslant f_i(x')$, and there is a $j \in \{1, \ldots, m\}$ such that $f_j(x) < f_j(x')$. A solution $x^\star \in \mathbb{R}^d$ is Pareto optimal if it there does not exist $x \in \mathbb{R}^d$ s.t. $x^\star$ is dominated by x. The set of all Pareto optimal solutions is the Pareto set; its mapping in the objective space is the Pareto front. Let $w \in \mathbb{R}^m$ be a weight vector, and $g(\cdot|w)$ be a scalarizing function that, given a solution, transforms its objective vector $F(x)$ into a scalar real value. For instance, one might want to use the Chebyshev function, defined by $g(x|w) = \max_i\{w_i \cdot |f_i(x) - z_i^*|\}$, where z^* is a reference point. Other scalarizing functions can be found in the MOEA literature [26].

To identify a good Pareto set approximation, a standard decomposition approach considers to solve a number of sub-problems, defined using different weight vectors for configuring the scalarizing function. The output is hence made up with different solutions,

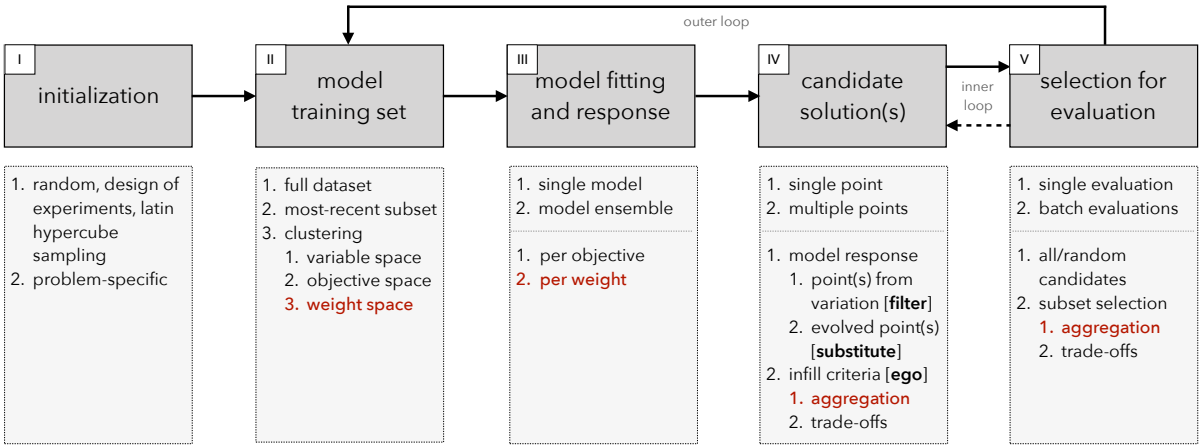

Figure 1: A refined taxonomy inspired from [7] and targeting a fine-grained classification of S-MOEAD. The text in colour correspond to algorithm components tailored to decomposition-based MOEAs.

corresponding to the defined sub-problems. Sub-problems may be solved independently [8] or cooperatively [26]. Besides, different evolutionary mechanisms can be adopted to evolve the solution set [24]. In an *expensive* setting, and looking at how surrogates can be combined with decomposition, most approaches share the use of a scalarizing function. However, different components and strategies are adopted to find good solutions w.r.t. the scalarized sub-problems in a restricted amount of evaluations. This makes it relatively difficult to differentiate between existing approaches and to identify their key components at first sight. In the next section, we describe a simple fine-grained classification of S-MOEAD in an attempt to capture their algorithmic gestalt.

2.2 A Fine-grained Classification of S-MOEAD

2.2.1 General outlook. The considered classification is summarized in Fig. 1. It is closely related to the taxonomy provided in [5], and is to be viewed as a fine-grained refinement targeting a better understanding of S-MOEAD. In fact, most surrogate-assisted MOEAs are following seemingly the same general template with basically four *a priori* interdependent layers, in addition to the initialization step. Initialization is particularly important in an expensive setting, since every single evaluation counts. Like all surrogate-assisted approaches, an S-MOEAD algorithm has to specify the set of initial points. This is not only of crucial importance for the search process, but also for the learning process, as the population serves as a pool from which the training set is constructed. Most algorithms use a Latin Hypercube Sampling (LHS), bare of any problem-specific heuristic. The general idea captured by the other layers is then as follows. Given a set of points from which the actual objective values are known, a pre-processing mechanism is used to prepare the data for training. Having built the training set, one or several models are fitted for a well-defined model response. These models are intended to help the search process by guiding the sampling and evaluation of new solutions in the subsequent layers. It is important to understand that the generation of new candidate solutions and the selection of solutions for evaluation are two different aspects that should be distinguished. In fact, a critically-important step is to generate a set of candidate solutions explicitly based on the surrogate models, or on the contrary implicitly by allowing the surrogate models to interfere with the conventional evolutionary search process. This is finally followed with a selection step that decides which subset of solutions is to be evaluated using the actual expensive objectives. We argue that the specification of these layers and the setting of their components allows one to differentiate a particular approach form another. It is worth noticing that the initial taxonomy provided in [7] does not include the training set construction phase, which will be shown to be a key ingredient in different S-MOEAD approaches. In the following, we further discuss the different layers in a more fine-grained manner, before turning to a more focused discussion of existing techniques.

2.2.2 Defining the model training set. This phase can take diverse forms, as almost each existing algorithm adopts a different approach. It is crucially important since the input data can highly bias the accuracy of the constructed model and/or influence the interpretation of any information extracted. A straightforward strategy, not necessarily specific to S-MOEAD, is simply to include all solutions evaluated so far as input for the model training/fitting and for the subsequent model-specific treatments. However, this might lead to the construction of a global surrogate at a high computational cost, eventually dominate the evaluation cost itself. A common alternative is to use only recently-evaluated points, that can be selected, e.g., over a predefined time window, or also based on the quality of the observed objective or scalar values. Another alternative is to construct the training set using a clustering procedure. This is a natural outcome in S-MOEAD approaches, since dealing with different sub-problems suggests to focus on specific target search regions directed by the value of the weight vector, and hence to clusterize the data accordingly. Interestingly, this could imply the use of a structured ensemble of surrogates used in the other layers to assist the optimization process. As such, we can find a number of data clustering techniques, ranging from the ones operating in the variable space, to those centered on the objective values of the points sampled so far, or even on the aggregated values defined with

respect to the scalarizing function and/or its weight vector. From a pure machine learning perspective, choosing/clustering the evaluated solutions is both related to the degree of locality expressed by the trained model and to the possibility of obtaining different trade-offs between the model quality and the computational cost of the fitting phase [21].

2.2.3 Model fitting and response. At this stage, one is concerned with the output prediction provided by the surrogate model, i.e. the response variable. A number of variants can be considered, depending on the type of training data and on the surrogate model by itself. The most straightforward approach is to fit the objective values, meaning that one model is fitted w.r.t. each objective. The scalarizing function can then be used to aggregate the predicted values. The other common approach is to fit the aggregated values, and an ensemble of models is hence constructed w.r.t. the weight vectors of the scalarizing functions. This might then be tightly coupled with a specific choice of the training set, eventually leading to a complex design. Besides, since different regression models and different hyper-parameters can be considered, different models can actually be trained concurrently and their responses exploited afterwards.

2.2.4 Candidate solution(s) generation. This phase consists in computing a set of solutions that are expected to be the most beneficial at the current state of the search process. This is critically important, since it specifically informs about the most promising regions that should be effectively searched. We identify three main aspects to this phase. First, one can target candidate solutions optimizing explicitly the model response, i.e. the objective values or their aggregation as predicted by the surrogate model(s). In this case, a standard evolutionary algorithm (EA) can for instance be carried out, while using the model response as a substitute for the real objective or scalarizing functions.Moreover, we differentiate between approaches running the EA for multiple iterations, e.g., run a whole MOEAD up to some predefined number of generations, or those running a single EA iteration before resuming, e.g., generate multiple offspring using standard variation operators and then choose a subset as candidate solutions. The former approach is usually termed as *substitute approach* [16], whether the latter is usually termed as *filtering* or pre-screening approach [14]. Second, one can target candidate solutions optimizing other criteria that are intrinsically linked with the surrogate model. This is typically the case of Gaussian processes, where a number of infill criteria are well established, e.g., Expected Improvement (EI), Probability of Improvement (PI), or Lower Confidence Bound (LCB) [1, 6, 11]. However, since we are dealing with a MOP, one could for instance search for a single solution optimizing the infill criteria with respect to one or multiple weight vectors, or run a standard MOEA where the objectives are defined as the infill criteria corresponding to each original objective. Finally, these different design choices can be guided by whether a single candidate solution is to be evaluated, or if a batch of solutions can be evaluated in parallel. Further considerations on this aspect are discussed in [5], not necessarily within the context of decomposition.

2.2.5 Selection for evaluation. When multiple candidate solutions are generated, a (multiobjective) subset selection mechanism is to be designed in order to prepare a batch of a desired size on which the true objective values is to be supplied. In the context of S-MOEAD, this can be done on the basis of the scalar values retrieved or computed from the model response, or based on the infill criteria, in addition to other options available from the literature. On the contrary, when a single candidate solution is generated while a batch of solutions is expected, one has to iterate over the previous phase, eventually injecting some knowledge about the solutions that are already contained in the batch. Once the batch is ready, the selected solutions are evaluated, and the whole process is repeated until some termination condition is satisfied. A maximum number of evaluations is typically adopted as a stopping condition.

2.3 Considered S-MOEAD Algorithms

Based on the previous discussion, we are able to classify a number of existing S-MOEAD techniques from the literature, and to differentiate them in a more systematic manner, as depicted in Table 1. Apart from the four first approaches, which follow a self-designed simple filtering archetype framework, the remaining one are based on more sophisticated components. Notice that all these approaches can be correctly classified according to the different layers provided in Fig. 1. As one can see, the considered approaches provide a representative combinations of the possible design choices. In the following, we shall describe them in more details and clarify any important hidden technical detail, since we also aim at conducting a comparative analysis of their behaviors in the next section.

2.3.1 S-MOEAD based on filtering. The approaches based on filtering are conceptually among the most basic S-MOEADs; see, e.g., [13]. Instead of generating one single solution by means of variation operators, a pool of $\lambda > 1$ solutions are generated. This pool constitutes the set of candidate solutions from which a batch has to be selected for evaluation. Once the evaluation is carried out, the training set is updated, the model is re-trained, and the next iteration of the original MOEAD can be executed. It should be clear that, even for this very simple archetype, different variants could be designed on the basis of the previously-discussed taxonomy. Interestingly, this archetype allows us to freely investigate the impact of setting the components from other layers, such as the impact of clustering or the response variable of the surrogate. We thereby consider four simple filter-based variants, based on the conventional MOEAD, as depicted in Table 1. The two first variants employ one surrogate model per objective, whereas the third and fourth ones train one surrogate model for each sub-problem defined by the weight vectors. The first and fourth variants use the whole set of solutions evaluated so far for training, whereas the second and third ones use two different clustering techniques for constructing the training set. The second variant performs a fuzzy clustering w.r.t. the position of solutions in the variable space, and train one model per cluster and per objective. In MOEAD-filter2, the estimation is based on the model linked to the cluster with the closest center to the considered solution. On the contrary, MOEAD-filter3 performs a clustering w.r.t. the aggregated values, each cluster being used for a given weight vector. Taken separately, these two clustering approaches are not our own proposal, since they are inspired by key components used in more advanced (state-of-the-art) non-filtering based S-MOEAD [9, 27]. The detailed description of these clustering techniques will be presented in more details below.

Table 1: S-MOEAD techniques as instances of the general framework.

algorithm	II – model training set	III – model fitting and response	IV – candidate solution(s)	V – selec. for evaluation
MOEAD-filter1	full dataset (1)	1 model per objective (1.1)	1 MOEAD iteration (2.1.1)	single, w.r.t. cur weight (1.2.1)
MOEAD-filter2	fuzzy clustering [27] (3.1)	1 model per cluster and objective (1.1)	1 MOEAD iteration (2.1.1)	single, w.r.t. cur weight (1.2.1)
MOEAD-filter3	weight-centered clustering [9] (3.3)	1 model per weight (1.2)	1 MOEAD iteration (2.1.1)	single, w.r.t. cur weight (1.2.1)
MOEAD-filter4	full dataset (1)	1 model per weight (1.2)	1 MOEAD iteration (2.1.1)	single, w.r.t. cur weight (1.2.1)
MOEAD-RBF [25]	best per weight (3.2)	3 models (kernels) per objective (2.1)	1 full MOEAD on model response (2.1.2)	batch, w.r.t. cur weights (2.2.2)
MOEAD-EGO [27]	fuzzy clustering (3.1)	1 model per cluster and objective (1.1)	1 full MOEAD on EI (2.2.2)	batch, w.r.t. cur weights (2.2.2)
M-EGO [9]	weight-centered clustering [9] (3.3)	1 model per weight (2.2)	1 full scalar EA on EI (1.2.1)	single, w.r.t. cur weight (1.2.1)
ParEGO [12]	best sol. for considered weight (3.3)	1 model for 1 weight (1.2)	1 full scalar EA on EI (1.2.1)	single, w.r.t. cur weight (1.2.1)

2.3.2 S-MOEAD based on substitution. A second class of S-MOEAD archetypes is based on using the surrogate as a substitute, by temporarily relying on its response for searching improving solutions. In [16], a conventional MOEAD optimizing the model response is executed repeatedly for a number of generations before a batch of evolved solutions is evaluated using the expensive objective functions. However, at each MOEAD iteration, i.e., when looping over weight vectors, a filtering approach selects the solution considered for the replacement phase among a number of offspring generated by variation. Another conceptually more sophisticated variant based on substitution is described in [25]. The so-called MOEAD-RBF constructs the training set using the best subset of evaluated solutions w.r.t. the weight vectors. Next, not one, but three radial basis function (RBF) surrogate models per objective are trained using different kernels. After training, the objective values from solutions evaluated so far are compared against the corresponding predicted values given by each of the so-constructed RBF model. The observed error is used to rank the accuracy of each model; see [25] for details. Then, a standard MOEAD is run for a number of generations using the RBF models as a substitute, but where the predicted values are weighted by a linear combination of the previously-computed model weights. The approximation set returned by MOEAD is then used as the set of candidate solutions from which a batch of solutions is to be selected. This is performed by considering a number of weight vectors from which a desired fraction is considered in a round-robin fashion. The candidate solutions having the best model-weighted aggregated values for the selected weight vectors is considered for the expensive evaluation.

2.3.3 S-MOEAD based on Gaussian processes. Many surrogate-assisted MOEAs are based on the extension and adaptation of the popular EGO (Expensive Global Optimization) procedure from single-objective optimization [10]. Existing S-MOEADs stand for no exception. EGO builds a Gaussian Process (GP) model of a function, where typically each point of the search space is viewed as a random variable that is assumed to follow a normal distribution. The generation of promising candidate solutions is fully guided by the knowledge that can be extracted from the model response surface. Different merit functions and infill criteria can be designed for scalar (single-objective) functions, such that the widely acknowledged Expected Improvement (EI) [10]. In the following, we review three reference EGO-like S-MOEADs and discuss their specific design components, in light of our classification.

ParEGO [12] is among the very first extensions of EGO for MOPs. It is based on the aggregation of the objectives using an augmented Chebyshev scalarizing function. At each iteration, one weigh vector is thrown randomly, and a single-objective EGO is considered for the so-defined scalar sub-problem. A specific design choice is to train the corresponding model using only the n (user defined) best-ranked solutions w.r.t. their aggregation values. It should be clear that generating weight vectors at random may be problematic, which in our opinion is the major reason motivating several subsequent S-MOEAD algorithms.

A sophisticated extension of EGO in combination with MOEAD can be found in [27]. The so-called MOEAD-EGO uses a fuzzy clustering procedure to build multiple clusters of solutions evaluated so far. The size of each cluster is set to a user-defined value k. Each cluster is used to build one surrogate model w.r.t. to each objective, i.e., $(k \cdot m)$ models are built at each iteration. A set of candidate solutions are then generated by means of an MOEAD execution to optimize simultaneously the EIs w.r.t. objective. However, it is shown that, when considering a single target weighted scalarizing function, a local surrogate model can be inferred analytically from the initial (non-aggregated) models. The aggregated EIs can then be inferred as well, which makes the optimization process more effective. More specifically, once the aggregated EI values of a candidate solution is required by the MOEAD inner optimizer, the surrogate model corresponding to the cluster with the closest center in the variable space is considered.

More recently, a seemingly different and high-performing EGO-like S-MOEAD was proposed [9]. Contrary to ParEGO, M-EGO considers a pre-defined set of weight vectors, corresponding to different sub-problems. It iterates over the weight vectors following a well-defined sequence. At a given iteration, considering a reference weight vector, the training set is chosen as the n closest points to that weight vector, using the projected euclidean distance of all solutions evaluated so far. A local surrogate for the target search direction is then built accordingly, and the optimization of the infill (EI) criteria is performed by means of a scalar EA.

3 EXPERIMENTAL ANALYSIS

3.1 Experimental Setup and Methodology

Apart from the 8 surrogate-assisted approaches listed in Table 1, we consider 3 surrogate-less algorithms, namely a standard MOEAD, a random search (RS), and a simple Latin Hypercube Sampling (LHS). In all MOEAs, the population size is set to 50, and is initialized by means of LHS. The scalarizing function is Chebyshev, whose reference point is updated with the best objective values seen so far. For filtering algorithms, we use the same parameters as in the MOEAD. At each filtering step, 8 offsprings are generated the one with the best aggregated predicted value for the current weight is evaluated. We measure the performance of the competing algorithms under different scenarios in terms of budget, measured as

Table 2: 12 considered bi-objective benchmark functions.

objective 1			objective 2				
			f_1	f_8	f_{14}	f_{15}	f_{20}
f_1	Sphere	separable	✓	✓	✓	✓	✓
f_8	Rosenbrock	moderate		✓	✓	✓	✓
f_{14}	Sum of powers	ill-conditioned			✓		
f_{15}	Rastrigin	multi-modal				✓	
f_{20}	Schwefel	weakly-structured					✓

a number of function evaluations: ranging from a tight budget of 500 evaluations to a larger budget of 2 500. For MOEAD-RBF and MOEAD-EGO, the number of evaluations *per* generation is 10. All algorithms are implemented in Python, and the experiments are conducted on a Intel(R) Core(TM) i5-5200U CPU with 2.20 GHz and 8 GB of RAM. We rely on the scikit-learn library [17]. while matching the configuration described in the original paper of each algorithm. For filtering approaches, we use a support vector regression (SVR) with a radial basis function (RBF) kernel. The size of the training set is bounded by 100 when clustering is applicable. It is made of all solutions evaluated so far otherwise.

In terms of problems, we consider a number of numerical functions extracted from the bi-objective black-box optimization benchmarking test suite (bbob-biobj) [15]. More particularly, we selected 12 functions with different properties in terms of separability, conditioning, multimodality, and global structure. They are listed in Table 2. In order to measure the ability of the considered approaches to cope with the problem dimensionality, we experimented functions with $d \in \{2, 3, 5, 10\}$ decision variables. We perform 10 independent runs for each problem and algorithm (5 280 experiment).

The algorithms are compared in terms of the hypervolume relative deviation to the best-found approximation set for the considered function. The hypervolume [2] measures the area covered by an approximation set and enclosed by a reference point. Due to the target of our experimental analysis, we consider two different settings for the reference point: a "global" reference point that maps to the worst objective value observed on a given instance once the algorithm starts at 50 evaluations, and a "local" reference point that maps to the worst objective value observed on all Pareto set approximations found for a given budget. that for Table 3, the rank obtained are the same for the 2 hypervolume considered.

3.2 Overall Approximation Quality

We first summarize our findings using two scenarios in terms of the global budget: a particularly tight setting of 500 evaluations, and a moderate setting of 2 500 evaluations. Table 3 reports the rank of every algorithm for each pair of functions and problem dimension. Given that 10 algorithms are compared, ranks range from 0 to 9, a lower value being better. The average rank of a given algorithm over all problems and dimensions is reported in the last row. A local hypervolume reference point is used, but we could check that this remains consistent when using a global reference point.

The strategies based on random search (RS) and latin hypercube sampling (LHS) never obtain a better rank than any other approach. Regarding the conventional (surrogate-less) MOEAD, it is almost always outperformed by the other S-MOEAD for a tight budget. However, when the budget is larger, its performance increases, especially when problem dimension is small. It obtains better rank than MOEAD-filter1, M-EGO and ParEGO when the number of evaluations is 2 500. When comparing the different S-MOEAD algorithms, the EGO approaches outperform filtering for a low budget, whereas the opposite holds for a larger budget. Hence, EGO approaches perform better at the early stages of the search process, while filtering approaches seem to slightly more time to converge. More about the convergence profile will be discussed later. Therefore, depending on the budget that can be afforded for the optimization process, one class of algorithms shall be preferred over the other. This highlights the critical importance of the optimization scenario and the careful considerations that must be taken into account while considering the expensive nature of the problem to be solved. It is also interesting to notice that the relative performance of EGO approaches is particularly good for large dimensional problems. For instance, ParEGO is never outperformed by any other approach apart from MOEAD-RBF for $d = 10$, which is still to be confirmed with even higher budgets. At last, we can highlight the superiority of MOEAD-RBF for both budget scenarios. Indeed, it significantly outperforms all other approaches on most instances, with an average rank of 0.4 and 0.2 respectively for 500 and 2 500 evaluations, and is never outperformed by more than two algorithms.

3.3 Impact of Clustering

In this section, we investigate the impact of clustering on the performance of S-MOEAD approaches. We recall that clustering enables to select solutions for constructing the training set. An obvious reason for using clustering is to reduce the computational overhead induced by model training, i.e., the larger the training set, the longer the training phase [12]. On the other side, it is often argued that a larger training set improves the overall model accuracy. However, it remains unclear how clustering actually impacts approximation quality which motivates our further investigations.

Let us focus on filtering approaches, by comparing first MOEAD-filter1, where no clustering is performed and where the models learn the objective functions, to MOEAD-filter2, also learning the objective functions but performing a fuzzy clustering. Similarly, we compare MOEAD-filter4, where no clustering is performed and where the models are learning scalarized values, to MOEAD-filter3, where the model also learns from scalarized values, but where a weight-centered clustering is performed. Bear in mind, that beside the clustering step, MEOAD-filter1 (resp. MOEAD-filter4) is identical to MOEAD-filter2 (resp. MOEAD-filter3). We should also note that the training set for MOEAD-filter1/4 is made of the entire set of solutions evaluated so far, whereas the cardinality of the training set in MOEAD-filter2/3 is bounded by 100.

As reported in Table 3, the relative rank of the four algorithms is almost the same for 500 evaluations. We attribute this to the fact that the difference in the training set size is still relatively small at the early stages of the search process. However, for a larger budget of 2 500 evaluations, the difference starts to show, and MOEAD-filter1 (resp. MOEAD-filter4) is clearly outperformed by MOEAD-filter2 (resp. MOEAD-filter3). In both cases, the algorithms using clustering manage to outperform their respective counterpart. This means that not only clustering allows for a faster computational

Table 3: Comparison of the competing algorithms with respect to the hypervolume relative deviation under a local reference point for 500 (left) and 2 500 (right) calls to the evaluation function. The rank stands for the number of algorithms that statistically outperform the one under consideration w.r.t a Mann-Whitney test with a p-value of 0.05 and a Bonferroni correction (lower is better). Bold values correspond to the best-performing algorithm for the problem under consideration.

		RS	LHS	MOEAD	MOEAD-filter1	MOEAD-filter2	MOEAD-filter3	MOEAD-filter4	MOEAD-RBF	MOEAD-EGO	M-EGO	ParEGO	RS	LHS	MOEAD	MOEAD-filter1	MOEAD-filter2	MOEAD-filter3	MOEAD-filter4	MOEAD-RBF	MOEAD-EGO	M-EGO	ParEGO
		#eval = 500											#eval = 2500										
d = 2	f1-f1	4	6	4	4	4	1	4	**0**	1	1	1	8	8	3	3	**0**	**0**	1	**0**	3	3	4
	f1-f8	6	4	2	3	1	2	**0**	**0**	**0**	1	4	7	7	**0**	2	**0**	**0**	**0**	**0**	**0**	5	7
	f1-f14	5	7	3	3	3	**0**	3	**0**	**0**	**0**	3	8	8	1	5	**0**	**0**	**0**	1	**0**	1	6
	f1-f15	4	4	3	3	**0**	1	**0**	1	**0**	**0**	1	7	7	**0**	1	1	**0**	**0**	1	**0**	3	4
	f1-f20	6	4	3	**0**	3	**0**	1	**0**	**0**	1	**0**	8	7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	5	5
	f8-f8	7	8	4	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	8	8	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	6
	f8-f14	6	7	1	**0**	1	**0**	1	**0**	**0**	1	1	6	6	**0**	**0**	2	**0**	**0**	**0**	**0**	3	4
	f8-f15	7	7	4	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	9	7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	f8-f20	7	5	3	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	8	7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	4	6
	f14-f14	6	9	4	3	3	**0**	**0**	**0**	**0**	**0**	**0**	8	8	2	2	**0**	**0**	**0**	2	4	2	4
	f15-f15	6	5	3	3	1	1	**0**	2	**0**	**0**	**0**	9	9	2	2	2	**0**	**0**	2	**0**	2	7
	f20-f20	7	4	4	**0**	1	**0**	**0**	**0**	**0**	**0**	1	9	9	2	1	**0**	**0**	**0**	2	1	5	6
d = 3	f1-f1	6	5	4	3	4	3	3	**0**	**0**	**0**	2	8	8	3	4	**0**	**0**	1	2	3	3	6
	f1-f8	6	5	1	**0**	1	1	1	**0**	**0**	1	1	9	5	1	**0**	2	**0**	2	**0**	**0**	2	3
	f1-f14	4	5	3	3	1	1	1	**0**	**0**	1	1	8	8	1	5	1	**0**	**0**	1	1	2	5
	f1-f15	7	8	3	1	2	2	2	**0**	**0**	**0**	1	7	7	1	1	**0**	**0**	**0**	1	3	3	6
	f1-f20	6	9	4	1	3	3	2	**0**	**0**	1	3	8	8	1	2	1	**0**	**0**	**0**	**0**	5	7
	f8-f8	6	8	3	1	3	3	2	**0**	**0**	**0**	**0**	9	9	1	1	**0**	**0**	**0**	1	3	1	3
	f8-f14	9	9	2	3	1	3	3	**0**	**0**	**0**	**0**	8	8	1	4	1	1	1	**0**	**0**	1	3
	f8-f15	6	8	5	3	3	**0**	3	**0**	**0**	**0**	**0**	9	9	1	**0**	**0**	**0**	**0**	1	2	1	4
	f8-f20	4	9	3	2	1	2	1	**0**	**0**	**0**	**0**	9	9	2	4	**0**	**0**	1	**0**	**0**	3	4
	f14-f14	8	8	3	1	1	2	2	**0**	**0**	**0**	1	8	8	1	1	1	**0**	1	1	1	2	6
	f15-f15	4	4	4	4	4	4	4	**0**	**0**	**0**	**0**	9	8	2	4	**0**	**0**	**0**	**0**	**0**	**0**	5
	f20-f20	8	8	4	4	3	3	3	**0**	**0**	**0**	**0**	9	9	3	2	**0**	**0**	**0**	2	3	2	5
d = 5	f1-f1	7	7	6	4	4	4	4	**0**	1	1	2	9	9	4	4	1	**0**	1	1	4	4	6
	f1-f8	9	8	4	4	4	4	4	**0**	**0**	**0**	2	9	9	1	1	3	1	1	**0**	1	1	5
	f1-f14	9	8	5	3	3	3	3	**0**	**0**	**0**	2	9	9	**0**	5	**0**	**0**	**0**	**0**	**0**	**0**	4
	f1-f15	5	6	5	3	4	4	3	**0**	**0**	**0**	1	8	8	1	5	1	**0**	1	**0**	**0**	1	1
	f1-f20	9	9	5	3	3	3	3	**0**	**0**	1	2	9	9	1	1	1	1	1	**0**	1	2	5
	f8-f8	8	8	8	4	4	4	4	1	1	**0**	**0**	9	9	1	1	**0**	**0**	**0**	**0**	1	**0**	**0**
	f8-f14	9	9	4	3	3	3	3	**0**	**0**	**0**	**0**	9	8	1	**0**	2	1	1	**0**	**0**	1	1
	f8-f15	8	8	8	3	3	3	3	**0**	**0**	**0**	**0**	9	9	4	4	**0**	**0**	**0**	**0**	**0**	1	3
	f8-f20	8	8	4	3	3	3	3	**0**	**0**	**0**	**0**	8	8	2	1	**0**	**0**	**0**	**0**	2	1	4
	f14-f14	9	9	3	3	3	3	3	**0**	3	**0**	**0**	8	9	1	6	2	**0**	1	**0**	7	1	2
	f15-f15	8	8	3	3	3	3	3	**0**	**0**	**0**	**0**	8	8	2	3	**0**	**0**	**0**	**0**	**0**	2	2
	f20-f20	8	9	8	3	3	3	3	**0**	2	**0**	1	9	9	1	7	**0**	**0**	**0**	1	3	1	4
d = 10	f1-f1	9	9	7	4	4	4	4	**0**	2	1	1	9	9	4	7	4	2	2	**0**	2	2	1
	f1-f8	9	9	7	4	4	4	4	**0**	1	1	1	9	9	3	3	3	3	3	**0**	1	1	1
	f1-f14	9	9	3	3	3	3	3	**0**	1	1	1	9	9	3	4	4	4	2	**0**	1	2	1
	f1-f15	8	8	5	3	3	3	3	**0**	1	1	1	9	9	3	6	3	3	3	**0**	1	2	1
	f1-f20	9	9	6	4	4	4	4	**0**	1	1	1	9	9	4	8	1	4	4	**0**	1	2	1
	f8-f8	9	9	8	4	4	4	4	1	2	1	**0**	9	9	5	7	4	3	1	**0**	1	2	1
	f8-f14	9	9	7	3	3	3	3	**0**	**0**	**0**	**0**	9	9	3	5	3	3	3	**0**	1	1	1
	f8-f15	9	9	7	3	3	3	3	**0**	**0**	**0**	**0**	9	9	3	3	3	3	3	**0**	1	1	1
	f8-f20	9	9	8	3	3	3	3	2	2	1	**0**	9	9	3	5	3	3	3	**0**	2	2	1
	f14-f14	9	9	4	3	3	3	3	2	**0**	**0**	**0**	9	9	3	3	3	3	3	**0**	1	1	**0**
	f15-f15	6	7	3	3	3	3	3	2	1	**0**	**0**	9	9	3	6	3	3	3	**0**	2	1	**0**
	f20-f20	9	9	7	4	4	4	4	**0**	2	2	**0**	9	9	4	8	4	2	4	**0**	2	2	1
	avg	**7.2**	**7.5**	**4.4**	**2.6**	**2.6**	**2.4**	**2.4**	**0.2**	**0.4**	**0.4**	**0.8**	**8.5**	**8.3**	**1.8**	**3.1**	**1.2**	**0.8**	**1**	**0.4**	**1.2**	**1.9**	**3.4**

time, as we will emphasize later, but it also helps to improve solution quality.

3.4 Convergence Profile

Convergence is a critical issue, especially in regards of the expensive setting we are interested in. Fig. 2 reports the evolution of the hypervolume relative deviation as the number of evaluations grows (for dimension $d = 5$). A close look at the relative convergence profile allows us to emphasize how much different are filtering approaches from EGO approaches. While the former seem to convergence almost log-log linearly in many cases, the latter reach a particularly good approximation quality just after about 250-500 evaluations, before getting stuck to a convergence point from which only minor improvements, if any, can be observed. Interestingly, the convergence profile of MOEAD-filter3 and MOEAD-filter4 seem to follow the same trend than the surrogate-less MOEAD, but with better hypervolume values all the way through. Ultimately, filtering approaches typically end up outperforming EGO approaches, after about 2 000 evaluations. Regarding MOEAD-RBF, it seems to conciliate the best of both worlds. In a similar way than EGO approaches, it converges particularly quickly at the early stages. However, it does not get stuck, and is able to continue improving at a slower convergence rate. In fact, its convergence profile dominates all others for all problems whose first objective is a Sphere function (f_1-$\star$), and for the most difficult considered problem (f_{20}-f_{20}).

Although we have no certainty as to why EGO approaches tend to get stuck at an early stage of the search process, a possible explanation might be the use of the expected improvement (EI) as infill criterion. Indeed, EI is used to discriminate solutions, with the aim of guiding the search process to the most interesting areas while enabling a good balance between solution quality and model refinement. Fig. 3 shows the EI value of the candidate solution

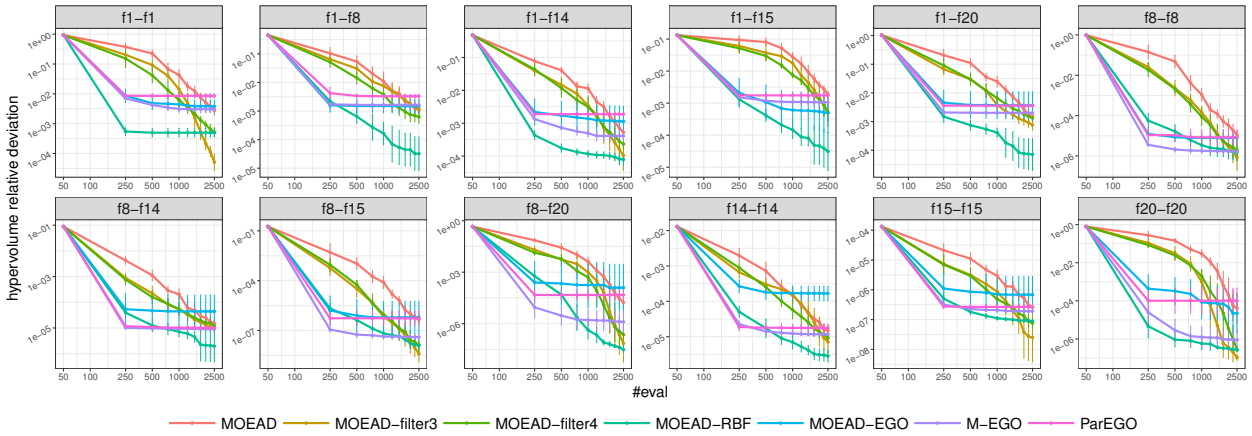

Figure 2: Convergence profile of the hypervolume relative deviation under a global reference point with respect to the number of expensive evaluations performed, for all problems of dimension $d = 5$. Notice the log scale on both axes.

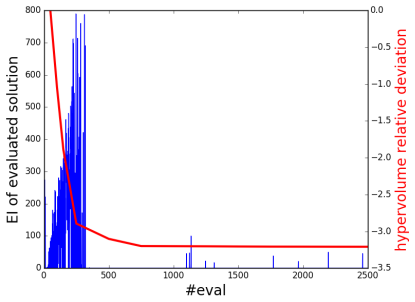

Figure 3: EI values of evaluated solutions and convergence plot for a typical run of ParEGO on f_1-f_1 with $d = 2$.

selected for evaluation at each step of a typical ParEGO run on the Sphere-Sphere problem for $d = 2$. Note that a similar trend is observed for other EGO approaches. We remind that EI values are positive, and the solution with the largest EI value is to be selected. Besides, once the model indicates that no improvement is expected, EI makes no difference between a descent and a poor solution, they will both have an EI value of 0. At the initial steps, the maximum EI value increases to a value close to 800 during the first 350 iterations, before abruptly dropping to values close to 0, apart from small peaks observed at isolated iterations. This means that, after some time, a lot of solutions actually have an EI of 0. By comparing this trend to the hypervolume relative deviation obtained over time, also reported in the figure, we clearly see the impact of EI values on the search progress. This seems to be a flaw of the EI infill criterion. As noted in [7], the performance of ParEGO could be improved using other infill criteria. This suggests that the EI measure can indeed be the source of the observed issue for EGO approaches. Further research on alternative infill criterion or some work-around for the observed issue, might lead to an improvement of EGO approaches.

4 OUTLOOK AND OPEN ISSUES

In this paper, we investigated surrogate-assisted MOEAs based on decomposition for expensive optimization. Based on a refined classification, we were able to instantiate existing approaches as well as new alternatives under a common framework, then highlighting their key design choices and components in a more systematic manner. A comparative analysis was conducted on a comprehensive benchmark of bi-objective black-box numerical optimization problems from the bbob-biobj test suite. Our main findings are as follows. Firstly, it appears clearly that carefully selecting solutions for training the model is of high importance, not only to reduce the amount of data for accelerating the training phase, but also to improve solution quality. Secondly, training multiple models implies to have an ensemble of surrogates. Combined with the design where the training set is clustered around weight vectors, surrogates are then likely to specialize to specific regions of the objective space, then matching the rationale of decomposition. As such, we argue that a more systematic investigation of this design choice would allow for explicit ways of designing and training specialized local surrogates. This is perfectly in line with the overall good performance of MOEAD-RBF, which also has the particularity of producing a whole batch of solutions to be evaluated. This is certainly of high importance in terms of computational complexity: not only this reduces the number of computationally-demanding model training tasks, this also enables the evaluation of multiple solutions at once in a parallel computing environment. The selection of the batch of solutions to be evaluated by the expensive objective functions shall then carefully cope with the multiobjective nature of the problem at hand by targeting diverse regions of the Pareto front. At last, the behavior of approaches based on Gaussian processes makes them very attractive for an extremely expensive optimization scenario, where the number of evaluations is reduced to the minimum. Notice that basic filtering approaches can be preferred when the application context can accommodate with a restricted but relatively moderate budget. Besides, the use of EI to select solutions seems to quickly inhibit the search progress, so that alternative infill criteria combined with adaptive and hybrid design choices of candidate solutions generation seems to be a promising future research line.

ACKNOWLEDGMENTS

This work was supported by the French national research agency (ANR-16-CE23-0013-01) and the Research Grants Council of Hong Kong (RGC Project No. A-CityU101/16).

REFERENCES

[1] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, 3:397–422, March 2003.

[2] Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. Theory of the hypervolume indicator: Optimal-distributions and the choice of the reference point. In *Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, FOGA '09, pages 87–102, New York, NY, USA, 2009. ACM.

[3] Juergen Branke, Kalyan Deb, Kaisa Miettinen, and Slowinski Roman. Multiobjective optimization, interactive and evolutionary approaches [outcome of dagstuhl seminars]. 01 2008.

[4] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya. A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 22(1):129–142, Feb 2018.

[5] K. Deb, R. Hussein, P. C. Roy, and G. Toscano-Pulido. A taxonomy for metamodeling frameworks for evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 23(1):104–116, Feb 2019.

[6] G. I. Hawe and J. K. Sykulski. Probability of improvement methods for constrained multi-objective optimization. In *2008 IET 7th International Conference on Computation in Electromagnetics*, pages 50–51, April 2008.

[7] Daniel Horn, Tobias Wagner, Dirk Biermann, Claus Weihs, and Bernd Bischl. Model-based multi-objective optimization: Taxonomy, multi-point proposal, toolbox and benchmark. In António Gaspar-Cunha, Carlos Henggeler Antunes, and Carlos Coello Coello, editors, *Evolutionary Multi-Criterion Optimization*, pages 64–78, Cham, 2015. Springer International Publishing.

[8] E. J. Hughes. Multiple single objective Pareto sampling. In *The 2003 Congress on Evolutionary Computation, 2003. CEC '03.*, volume 4, pages 2678–2684 Vol.4, Dec 2003.

[9] Rayan Hussein and Kalyanmoy Deb. A generative kriging surrogate model for constrained and unconstrained multi-objective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, pages 573–580, New York, NY, USA, 2016. ACM.

[10] Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, Dec 1998.

[11] W. Kleijnen; J.P.C.; van Beers and van Nieuwenhuyse. Expected improvement in efficient global optimization through bootstrapped kriging. *journal of global optimization*, 2012.

[12] J. Knowles. Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, Feb 2006.

[13] Yung Siang Liau, Kay Chen Tan, Jun Hu, Xin Qiu, and Sen Bong Gee. Machine learning enhanced multi-objective evolutionary algorithm based on decomposition. In Hujun Yin, Ke Tang, Yang Gao, Frank Klawonn, Minho Lee, Thomas Weise, Bin Li, and Xin Yao, editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2013*, pages 553–560, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[14] Ilya Loshchilov. *Surrogate-Assisted Evolutionary Algorithms*. PhD thesis, 2013. ThÃĺse de doctorat dirigÃľe par Schoenauer, Marc et Sebag, MichÃĺle Informatique Paris 11 2013.

[15] O. Mersmann; T. TuÅąar; N. Hansen; A. Auger and D. Brockhoff. Coco: A platform for comparing continuous optimizers in a black-box setting. *ArXiv e-prints*, 2016.

[16] L. M. Pavelski, M. R. Delgado, C. P. d. Almeida, R. A. GonÃğalves, and S. M. Venske. Elmoea/d-de: Extreme learning surrogate models in multi-objective optimization based on decomposition and differential evolution. In *2014 Brazilian Conference on Intelligent Systems*, pages 318–323, Oct 2014.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[18] Wolfgang Ponweiser, Tobias Wagner, Dirk Biermann, and Markus Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted s-metric selection. In Günter Rudolph, Thomas Jansen, Nicola Beume, Simon Lucas, and Carlo Poloni, editors, *Parallel Problem Solving from Nature – PPSN X*, pages 784–794, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[19] Jerome Sacks, William J. Welch, Toby J. Mitchell, and Henry P. Wynn. Design and analysis of computer experiments. *Statist. Sci.*, 4(4):409–423, 11 1989.

[20] M. Sagawa, H. Aguirre, F. Daolio, A. Liefooghe, B. Derbel, S. Verel, and K. Tanaka. Learning variable importance to guide recombination. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, Dec 2016.

[21] Edward Snelson and Zoubin Ghahramani. Local and global sparse gaussian process approximations. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 524–531, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR.

[22] H. Tamaki, H. Kita, and S. Kobayashi. Multi-objective optimization by genetic algorithms: a review. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 517–522, May 1996.

[23] Yoel Tenne and Chi-Keong Goh. *Computational Intelligence in Expensive Optimization Problems*. 01 2010.

[24] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, 21(3):440–462, June 2017.

[25] Saúl Zapotecas Martínez and Carlos A. Coello Coello. MOEA/D assisted by rbf networks for expensive multi-objective optimization problems. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, pages 1405–1412, New York, NY, USA, 2013. ACM.

[26] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, Dec 2007.

[27] Q. Zhang, W. Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by MOEA/D with gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14(3):456–474, June 2010.

[28] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, Nov 1999.